System Requirements Document for Source Code Analysis Tool

Oğuzhan Yangöz, Ashwin Kishore, Yitao Wei

CSCI 6231 Software Engineering

March 11, 2022

1. Preface

This System Requirements Document (SRD) was developed to provide documentation and guidance to the Source Code Analysis Tool project development team. Its target audience is as follows;

a) current software developers, engineers and managers working on Source Code Analysis Tool project,
b) future software developers, engineers and managers who will join the development team of Source Code Analysis Tool project.
c) The clients who are the intended user of the project who shape the project by expressing their needs and funding the development.

1.1 Version History

Version 1.0
This is the first version of the document, and therefore, no version history is available. This section shall be updated with all the changes as the newer version of the document is released.

1.2 Definition and Acronyms

SCA: Source Code Analysis –the project being developed that is discussed in this document.

SRD: System Requirements Document – describes the user requirements, system requirements, functional and non-functional requirements of the SCA project (Kaisler, 2022).

SDD: System Design Document – describes "the system/software architecture, system/subsystem decomposition and communication protocols between systems/subsystems" (Kaisler, 2022).

STT: System Test Document –provides descriptions and steps of testing procedures for the software system to check whether the project meets the requirements in SRD.

Process: The process of interaction starting with the smallest element of a system to the larger components.

Object: An instance of an Object-oriented design, which represents real-world entity.

Class: A piece of extendable code template that is used for declaring variables, their initial values and methods for the intended behavior.

Attribute: A specification of class with a specific data type and structure.

Data type: A set of built-in types for variables such as Boolean, Float, Double, Boolean, Integer and Array.

Method: A set of operations that is executed upon a call through both global or local parameters.

Component: A collection of classes that provides a certain functionality of a system.

Module: A collection of components within a system or subsystem to provide certain functionality.

Subsystem: A large collection of modules to implement a large functionality. (e.g. Data Entry subsystem)

System: An organized collection of modules/subsystems that create a fully functional piece of software by interacting its subsystems efficiently in line with the customer requirements.

Repository: A container or storage that stores a chunk of data.

Session: The time elapsed between the last log-in and log-out of a user.

Account: The combination of a username and password that grants access a user to the Source Code Analysis tool.

User: A developer, engineer or manager that is responsible for developing and implementing the system.

Role: A comprehensive title that defines a set of responsibilities and authorizations of a user in the project such as Engineer, Developer and Manager.

2. Introduction

The demand for software development has gained substantial momentum in the last few decades worldwide. This rapid increase in the demand emerged a need for more complex systems that require collaborative work of multiple teams, making it hard to track and document the software development. This tool aims to assist developers, engineers, and managers in capturing and tracking data and information in large-scale software projects by encouraging all stakeholders to document each step of the software development. SCA keeps track of many elements including variables, classes, methods, functions, interfaces and hierarchical structures of a project. Furthermore, SCA ensures providing guidance to new team members, making maintenance and improvement of the projects through detailed descriptions, version history and UML diagrams.

3. User Requirements Definition

User requirements refer to the intended functionality set by the customers that shall be satisfied in the final version of the product. User requirements can be modified, replaced, or extended based on the customer needs. In this document, user requirements are listed under five major categories: User Interface, Query Subsystem, Report Subsystem, Data Entry Subsystem and Functional Analysis user requirements.

4.1 User Interface User Requirements

4.1.1. The system shall provide a unique username and password for each user.

4.1.2. The system shall allow users access the tool upon successful verification of user credentials.
4.1.3 The system shall display the main screen in Figure 2 upon user authentication.
4.1.4. The system shall grant users access to the command line interface on the main screen where users can interact with the repositories.
4.1.5. The system shall allow users work on multiple files within the same repository in different tabs concurrently, such as viewing class hierarchy model in one tab and viewing classes of a component in another tab.
4.1.6. The system shall allow users to view their recent session information and details, such as the duration of their current session, recently auto-saved versions of their work, and TO-DO items submitted by the managers.
4.1.7. The system shall allow users to open a repository through the File menu on the main screen.
4.1.8. The system shall grant users access to a toolbar on the main screen for making changes on files within a repository. The toolbar shall include the following options: undo the change(s), highlight the change(s), side-by-side comparison of two different versions of a file, make a comment/ annotation to other team members specifying a certain part of a file, dragging pre-specified templates for class hierarchy, new methods and flow chart.
4.1.9. The system shall allow users manage their user account within the boundaries of their assigned role in the organization.

4.2. Query Subsystem User Requirements

4.2.1 The system shall allow users make queries following the pre-defined query formats.
4.2.2 The system shall allow users make ad hoc queries.
4.2.3 The system shall allow users specify the maximum number of results they request in a query.
4.2.4 The system shall display users a message when they enter inaccurate input parameters.
4.2.5 The system shall display users an error message when they attempt to access a repository that does not exist.
4.2.6 The system shall accept a repository name and version as a parameter, if applicable.
4.2.7 The system shall allow users specify the fields in a query.

4.2.8 The system shall allow users specify the format of the queries they make.

4.3 Report Subsystem User Requirements

4.3.1 The system shall analyze all components of the system based on the repository to generate a detailed report.
4.3.2 The report shall be able to see the individual structure of an element of the system.
4.3.3 The system shall be able to export reports.
4.3.4 The system shall provide the capability to filter the contents of reports based on user-selected parameters

4.4 Data Entry Subsystem User Requirements

4.4.1 The system shall analyze the entered data in the repository based on command-based data entry.
4.4.2 The system shall analyze different selected types of data in the menu-selected forms.
4.4.3 The system shall analyze the name for updated objects the repository and pop up a window to edit the objects.
4.4.4 The system shall analyze attributes assigned to an object and pop up a window to edit the attributes.
4.4.5 The system shall analyze methods assigned to an object and pop up a window to edit the methods.
4.4.6 The system shall analyze specific information for every object in the repository and display it.
4.4.7 When the update is unsuccessful, the change will not be applied and pop up a warning window.

4.5 Functional Analysis User Requirements

4.5.1 The system shall allow users to keep track of decomposition in a relational UML structure.
4.5.2 The system shall allow users to access all the records that are required in the Data Entry System.
4.5.3 The system shall allow users to access all the attributes, their values and data types.
4.5.4 The system shall provide users with a description for each attribute.

4.5.5. The system shall grant users access to all the parameters from the methods invoked from other classes.

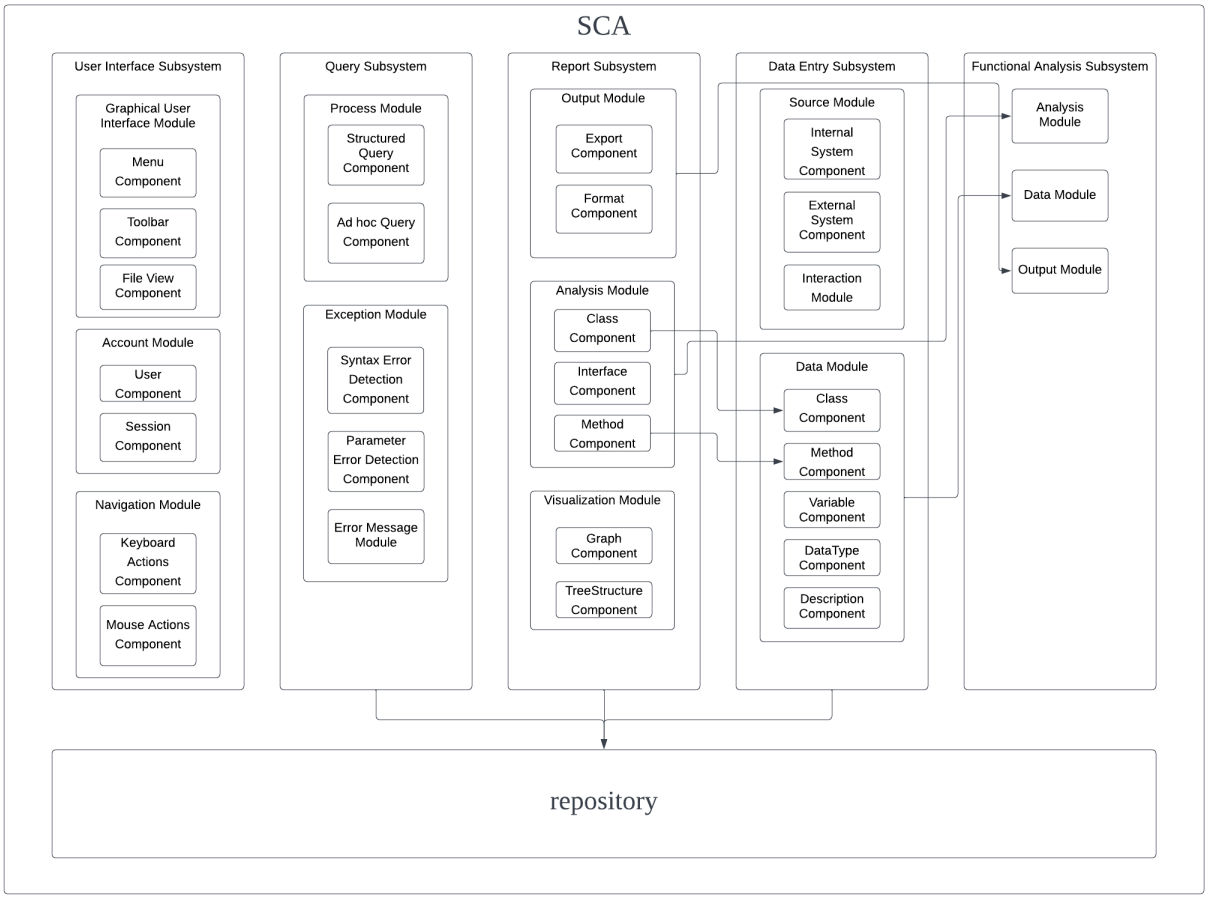
4.5.6 The system shall allow users to access a record of parent-child class relationship as a UML diagram.

4.5.7 The system shall allow users access to the return type of all methods.

4.5.8 The system shall allow users access to all the classes defined in the module.

4.5.9 The system shall allow users access the global values from all classes within the module.

5. System Architecture

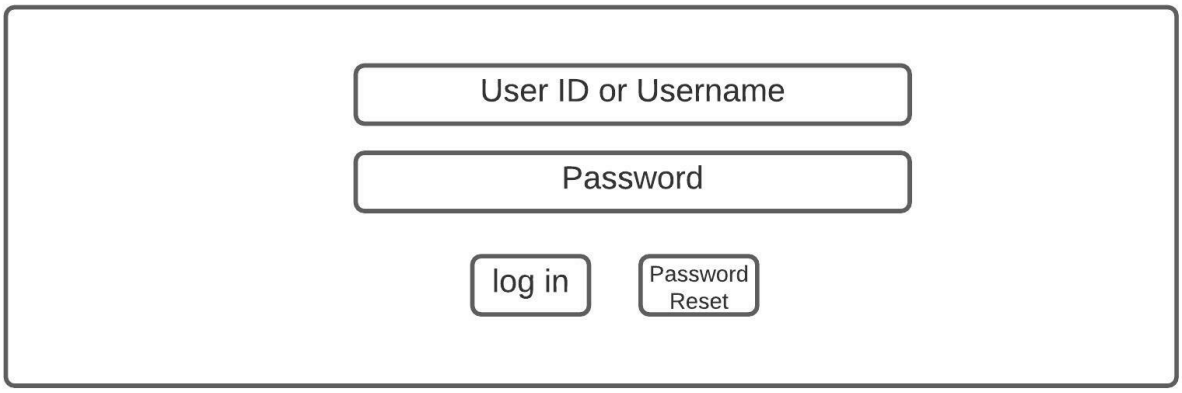


6. System Requirements

System Requirements refer to the functionality and behavior of a system or a program that is desired by the customers and implemented by the software development team. They clearly describe the expected outcome of a module or a subsystem in a testable way while addressing the user requirements in a coherent way. In this document, system requirements were listed under five categories: User Interface requirements, query subsystem requirements, report subsystem requirements, data entry subsystem requirements and functional analysis subsystem requirements.

6.1 User Interface Requirements

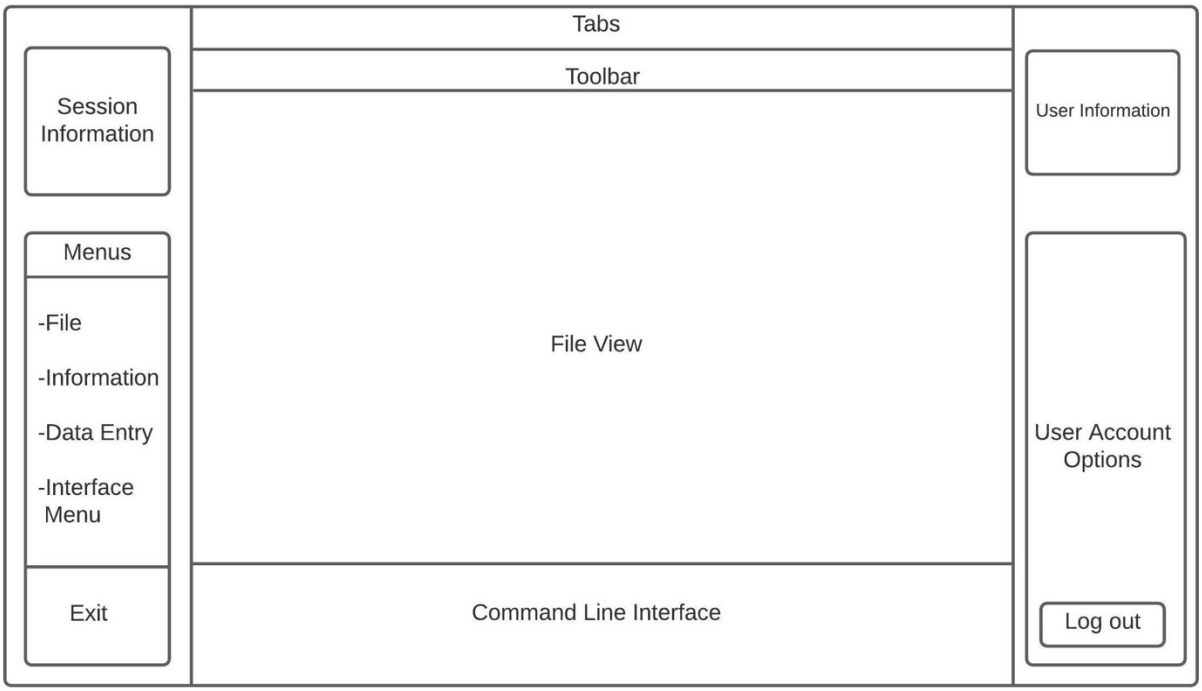
Figure 1: Log-in Panel



6.1.1 The SCA tool shall assign a unique username and password assigned to each user in its database.

6.1.2 The SCA tool shall have a login panel that allows each user to enter their credentials to access the tool. (See Figure 1- Log-in Panel for conceptual design that might be changed in the final version)

Figure 2: User Interface - Main Screen Layout



6.1.3 The SCA tool shall have a main screen that will be displayed every time a user is successfully logged in. (See Figure 2 - User Interface – Main Screen Layout for conceptual design that will be redesigned in the final version)

6.1.4 The SCA tool shall have a command line interface on the main screen when users are successfully logged in.

6.1.5 The SCA tool shall have logout and exit options on the main screen.

6.1.6 The SCA tool shall support multiple file views through tabs section (See Figure 2 - User Interface – Main Screen Layout). When a user clicks tab, File View shall display the content of that tab. The system shall allow side-by-side view of two files through the side-by-side functionality in the toolbar as well.

6.1.7 The SCA tool shall keep an entry for each session and display session information on the main screen. (See Definitions and Abbreviations section for “Session”)

6.1.8 The SCA tool shall display a window for users to open a repository when the File menu is selected.

6.1.9 The SCA tool shall have a toolbar for users which shall allow them to undo the change(s), highlight the change(s), see a side-by-side comparison of two different versions of a file, make a

comment/annotation to other team members specifying a certain part of a file, dragging a pre-specified templates for class hierarchy, new methods and flow chart onto the file in the File View screen.

6.1.10 The SCA tool shall have the user account options menu for account-related settings.

6.1.11 The SCA tool shall check whether the target repository exists and display a message if it doesn't.

6.1.12 The SCA tool shall autosave the changes to the repository's original directory every time a change is made.

6.1.13 The SCA tool shall save the changes before a repository is closed.

6.1.14 The SCA tool shall create an entry to store the directory or URL of each repository every time a new repository is created from scratch. This shall not apply to existing repositories as they shall be accessible by the users upon running the Source Code Analysis Tool.

6.1.15 The SCA tool shall display the logged in user's ID, account options and session duration on the main screen when users are successfully logged in.

6.2 Query Subsystem

6.2.1 The SCA tool shall have predefined multiple query formats for different searches.

6.2.2 The SCA tool shall allow queries without a predefined query format as well.

6.2.3 The SCA tool shall prompt users for specific search parameters and values of some fields for explicit queries.

6.2.4 The SCA tool shall prompt users for the maximum number of results requested.

6.2.5 The SCA tool shall check whether the type of the input parameters was correctly entered and display an error message if it was not.

6.2.6 The SCA tool shall check whether the repository contains the entities searched and print out an error message if not.

6.3 Report Subsystem

6.3.1 The Report Subsystem shall list all names for subsystems, modules, and classes by order and the lists are collapsed initially. If you want to see the details for a specific class, you should expand it.

6.3.2 The Report Subsystem shall list all interfaces by name when user specify a class.

6.3.3 The Report Subsystem shall list the attributes and methods of a class when user specify a class.

6.3.4 The Report Subsystem shall list the structure of a module in terms of classes..

6.3.5 The Report Subsystem shall list the structure of a subsystem in terms of modules.

6.3.6 The Report Subsystem shall list the structure of the system in terms of systems.

6.3.7 The Report Subsystem shall support several common report formats such as pdf, doc, and xls.

6.3.8 The subsystem shall provide the capability to uniform the structure of the report when users export report.

6.3.9 The Report Subsystem shall present function call sequences in the form of a tree.

6.3.10 Filter options include time, type of objects, and certain classes or function.

6.4 Data Entry Subsystem

6.4.1 The Data Entry System shall check if there is a repeat object in the repository when user attempt to create new object. If it has already existed, a warning window will pop up.

6.4.2 The user shall follow the certain format when assign a method to an object. An example of a specification is as follows:

<class name>[<Method name>, <method type>][<list of arguments by name>, <list of arguments by data type>]

6.4.3. The Data Entry Subsystem shall accept the following data types to the repository: System, Subsystem, Module, Class, Method, Attribute, Interface.

6.4.4 The subsystem shall be able to satisfy the interface requirements for both internal and external systems and communicate mutually.

6.4.5 The Data Entry Subsystem shall record which variables are used during system running.

6.4.6 The Data Entry Subsystem shall capture each variable, its data type, and description.

6.4.7 The descriptions of objects:

(a) Each object shall be described by the object name and the notes developer commented..

(b) Each object shall have a date and time stamp associated with its creation in the repository.

(c) Each object shall have a date and time stamp associated with the last time the object was updated.

(d) Each object shall have an identifier of the user who created the object in the repository. The user name is corresponding to who logged in to the system.
(e) Each object shall have an identifier of the last person to update the object in the repository. The user name is corresponding to who logged in to the system.

6.5 Functional Analysis Subsystem

6.5.1 The SCA tool shall keep a record of decomposition in a relational UML structure.
6.5.2 SCA tool shall capture and keep a record of parent-child class relationship as an UML diagram.
6.5.3 The SCA tool shall store all the records that are required in the Data Entry Subsystem.
6.5.4 The SCA tool shall keep a record of values and data types of all the attributes.
6.5.5 The SCA tool shall provide a description along with the attributes.
6.5.6 The SCA tool shall keep a record of all the parameters and their data types for all the methods.

7. System Models

It will be implemented later.

8. System Evolution

It will be implemented later.

9. Appendices

9.1 Hardware Requirements

Recommended hardware requirements for full functionality of the Source Code Analysis tool is as follows:
Operating System: MacOS Catalina 15.2 or higher OR Windows 10 or higher
Processor: 1.4 GHz Quad-Core Intel Core i5 or higher OR Apple Silicone M1 or higher
Graphics: Intel Iris Plus Graphics 645 1536 or higher
Memory: 8 GB 2133 MHz LPDDR3 or higher
Storage: 30 GB SSD/HDD or higher
Lack of any of this hardware may result in reduced functionality and performance.

10. Index

REFERENCES

Kaisler, S. (2022). *Requirements Engineering – I* [PowerPoint Slides]. School of Engineering and Applied Sciences, The George Washington University. https://blackboard.gwu.edu/bbcswebdav/pid-11929822-dt-content-rid-98309784_2/xid-98309784_2